Mathematical and Computational Applications, Vol. 10, No. 1, pp. 121-132, 2005.

DYNAMIC SYSTEM MODELLING USING RULES3 INDUCTION ALGORITHM

M.Sabih Aksoy
King Saud University, College of Computer and Information Sciences
P.O.Box 51178 Riyadh 11543 Saudi Arabia. msaksoy@ccis.ksu.edu.sa

Abstract- This paper describes the use of a new induction algorithm to derive production rules for modelling dynamic systems. The algorithm, called RULES-3, is an efficient tool for extracting a compact set of IF-THEN rules from a collection of examples to classify objects into known categories. The paper summarises the operation of RULES-3 and presents the results obtained in the modelling of two linear systems and two non-linear systems.
Key words- System Identification, Inductive Learning, RULES3.

1. INTRODUCTION

In general, to control a system which changes continually or for which a model cannot be derived through mathematical analysis, methods for obtaining a model of the system experimentally are required. Until recently, the most common methods have involved assuming a model structure and determining the values of the model parameters by experimentation (see, for example [1]). These methods rely on choosing a correct structure, which demands much experience and intuition from the control engineer. In the past few years, new machine learning techniques for experimental model building have been developed that do not need making assumptions about the structure of the model. These techniques can be separated into two types, neural-network-based techniques and induction-based techniques.

Neural-network-based techniques are simple and robust and have been successfully applied to the modelling of a variety of linear and non-linear systems [2;3;4; 5; 6; 7; 8; 9; 10, 11]. However, a disadvantage with these techniques is that the models produced only implicitly mimic the surface behaviours of the original systems, are non-transparent and therefore cannot be easily scrutinised or modified.

Induction-based techniques are also simple and reliable, but have the additional benefit of yielding more transparent models. Models obtained using induction are either represented as decision trees or production rules. Although they essentially will only mimic the external behaviours of the original systems, these models can be readily examined and modified if necessary.

The tool for induction-based modelling is the induction algorithm which extracts general patterns from specific examples. A number of induction algorithms exist, including CLS , ID3, ID4, ID5, BCT, AQ [12], C4.5, [13] and RULES family of algorithms [14; 15; 16].

This paper describes the use of RULES-3, the latest version of RULES, for inductive model building. Following a summary of the operation of RULES-3, the paper will present the results obtained with that algorithm in modelling four different systems.

2. RULES-3

RULES-3 [14] is a simple algorithm for extracting a set of classification rules from a collection of examples for objects belonging to one of a number of known classes. An object must be described in terms of a fixed set of attributes, each with its own range of possible values which could be nominal or numerical. For example, attribute "speed" might have nominal values {low, medium, high} or numerical values in the range [-10, 10].

An attribute-value pair constitutes a condition in a rule. If the number of attributes is N_a , a rule may contain between one and N_a conditions. Only conjunction of conditions is permitted in a rule and therefore the attributes must all be different if the rule comprises more than one condition. RULES-3 extracts rules by considering one example at a time. It forms an array consisting of all attribute-value pairs associated with the object in that example, the total number of elements in the array being equal to the number of attributes. The rule forming procedure may require at most N_a iterations per example. In the first iteration, rules may be produced with one element from the array. Each element is examined in turn to see if for the complete example collection it appears only in objects belonging to one class. If so, a candidate rule is obtained with that element as the condition. In either case, the next element is taken and the examination repeated until all elements in the array have been considered. At this stage, if no rules have been formed, the second iteration begins with two elements of the array being examined at a time. Rules formed in the second iteration therefore have two conditions. The procedure continues until an iteration when one or more candidate rules can be extracted or the maximum number of iterations for the example is reached. In the latter case, the example itself is adopted as the rule. If more than one candidate rule is formed for an example, the rule that classifies the highest number of examples, is selected and used to classify objects in the collection of examples. Examples of which objects are classified by the selected rule are removed from the collection. The next example remaining in the collection is then taken and rule extraction is carried out for that example. This procedure continues until there are no examples left in the collection and all objects have been classified. Figure 1 summarises the steps involved in RULES-3.

Step 1. Define ranges for the attributes which have numerical values and assign labels to those ranges
Step 2. Set the minimum number of conditions (N_{cmin}) for each rule
Step 3. Take an unclassified example
Step 4. $N_c = N_{cmin} - 1$
Step 5. If $N_c < N_a$ then $N_c = N_c + 1$
Step 6. Take all values or labels contained in the example
Step 7. Form objects which are combinations of N_c values or labels taken from the values or labels obtained in Step 6
Step 8. If at least one of the objects belongs to a unique class then form rules with those objects;
ELSE go to Step 5
Step 9. Select the rule which classifies the highest number of examples
Step 11. Remove examples classified by the selected rule
Step 12. If there are no more unclassified examples then STOP;
ELSE go to Step 3

Figure 1. Induction procedure in RULES-3 (N_c=number of conditions, N_a=number of attributes).

3. SYSTEM MODELLING EXPERIMENTS

The model building ability of RULES-3 was tested on four systems.

3.1. System 1

System 1 is a linear second-order system previously modelled using a neural network [17]. Its discrete input-output equation is:

$$y(t+1) = 1.8y(t) - 0.837y(t-1) + 0.019u(t) + 0.018u(t-1)$$

Thus the system has four essential attributes: *y(t), y(t-1), u(t)*, and *u(t-1)*. Two cases were tested. In one case, the range of input and output variables was divided into four equal intervals or quantisation levels. In the other case, nine quantisation levels were employed. This yielded 4 values per attribute in one case and 9 values per attribute in the other. 16 rules and 18 rules were extracted by RULES-3 for the first and the second cases respectively. The quantised responses of the system and the models to a step input are shown in Figures 2 and 3. As can be expected, the model was more accurate when the number of quantisation levels was higher, although in both cases the responses of the models closely followed those of the plant.

Figure 4 displays the responses of models with 5 attributes (y(t), y(t-1), y(t-2), u(t), u(t-1)) and 6 attributes (y(t), y(t-1), y(t-2), u(t), u(t-1), u(t-2)). Those models were obtained to test the ability of RULES-3 to handle situations where the order or structure of the system of interest is unknown. Apart from the model based on three

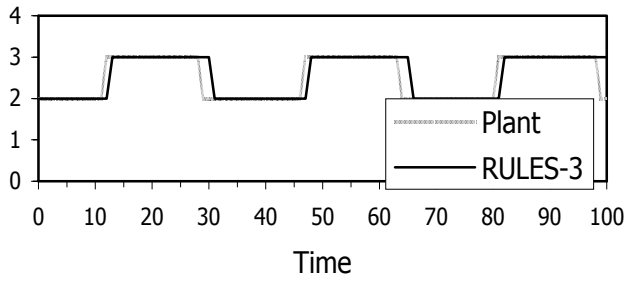

Figure 2. Quantised responses of System 1 and of its model obtained by RULES-3 (number of quantisation levels = 4).

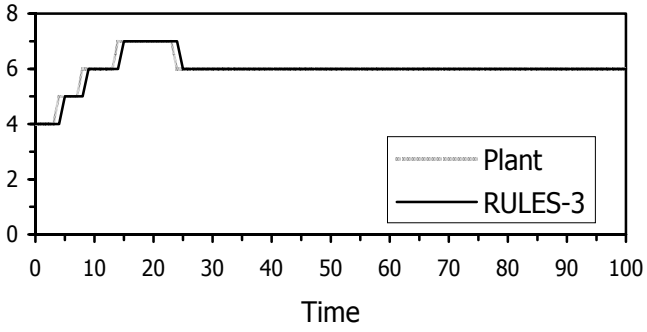

Figure 3. Quantised responses of System 1 and of its model obtained by RULES-3 (number of quantisation levels = 9; number of attributes = 4).

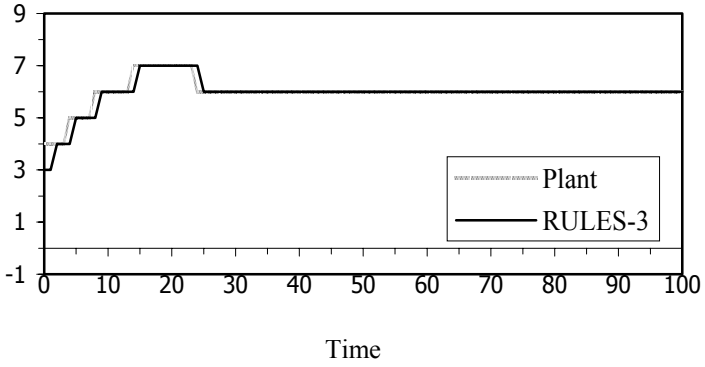

Figure 4. Quantised responses of System 1 and of its model obtained by RULES-3 (number of quantisation levels = 9; number of attributes = 5 or 6).

attributes (y(t-1), u(t), u(t-1)) (see Figure 5 for its response), which can only represent a system of lower order than the actual plant, the other models all gave responses almost identical to it. This demonstrates that the induction algorithm can derive good models without information on the exact number of attributes, or the system order, involved.

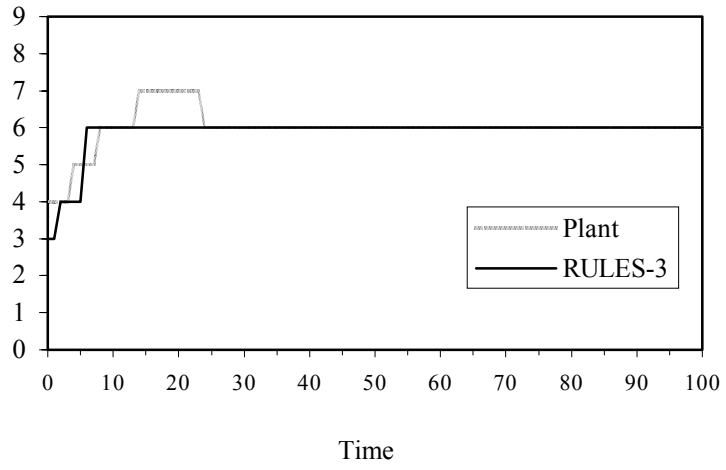

Figure 5. Quantised responses of System 1 and of its model obtained by RULES-3 (number of quantisation levels = 9; number of attributes = 3).

For comparison, the quantised response of a neural network using a multi-layer perceptron (MLP) with 4 input units, 19 hidden units and 9 output units is plotted in Figure 6. The neural network also modelled the plant well. However, its training time, at 23.11 minutes, was considerably longer than the model extraction time of 2.52 seconds taken by RULES-3.

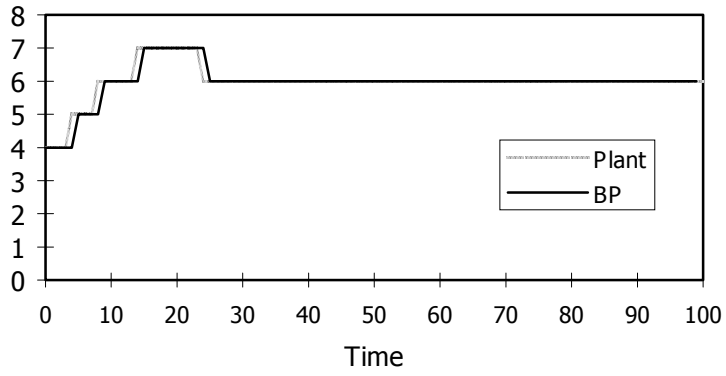

Figure 6. Quantised responses of System 1 and of its modelobtained by MLP. (number of quantisation levels = 9; number of attributes = 4).

3.2. System 2

System 2 is a non-linear plant with the following input-output equation:

$$y(t+1) = \frac{y(t)}{1+y^2(t)} + u^3(t)$$

u(t) and *y(t)* are the two essential attributes for this problem. Seven quantisation levels were each defined for *u(t)* and *y(t)*. 13 rules were extracted by RULES-3 as the model of the plant. A series of random signals was applied to the plant as shown in Figure 7

and the responses obtained are plotted in Figure 8. Note that the RULES-3 model behaved exactly as the plant for 67 of the 74 time instants shown. Only the data corresponding to the initial 18 time instants had been employed to extract the model. This shows the excellent generalisation ability of RULES-3.

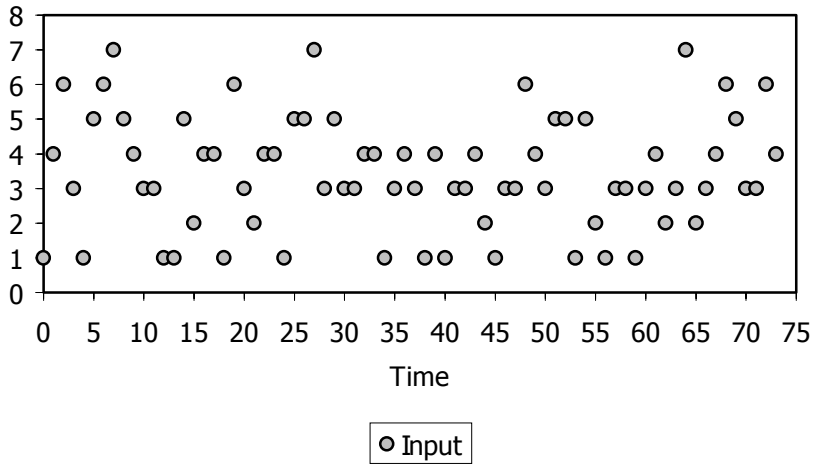

Figure 7. Quantised input to System 2 (number of quantisation levels = 7).

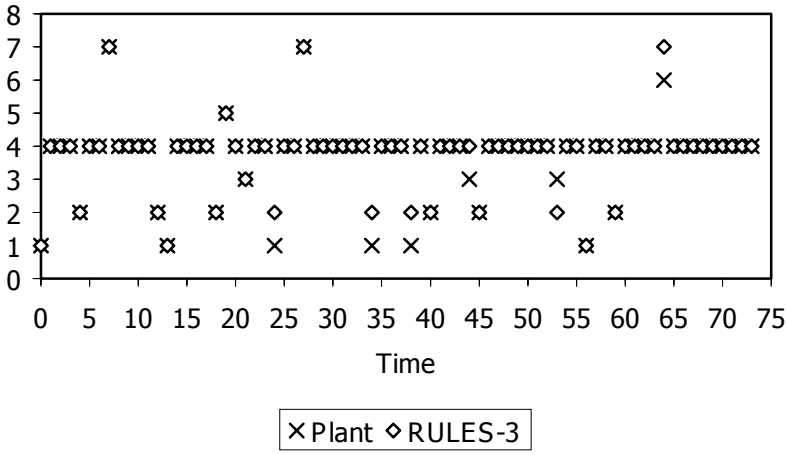

Figure 8. Quantised responses of System 2 and of its model obtained by RULES-3 (number of quantisation levels = 7; number of attributes = 2).

As with System 1, models were also generated without assuming prior knowledge of the plant structure. The responses of models based on three attributes (y(t), y(t-1), u(t)) and four attributes (y(t), y(t-1), u(t), u(t-1)) are plotted in Figures 9 and 10, superimposed on the expected responses of the actual plant. These show that even when irrelevant attributes were used, RULES-3 could still produce accurate models.

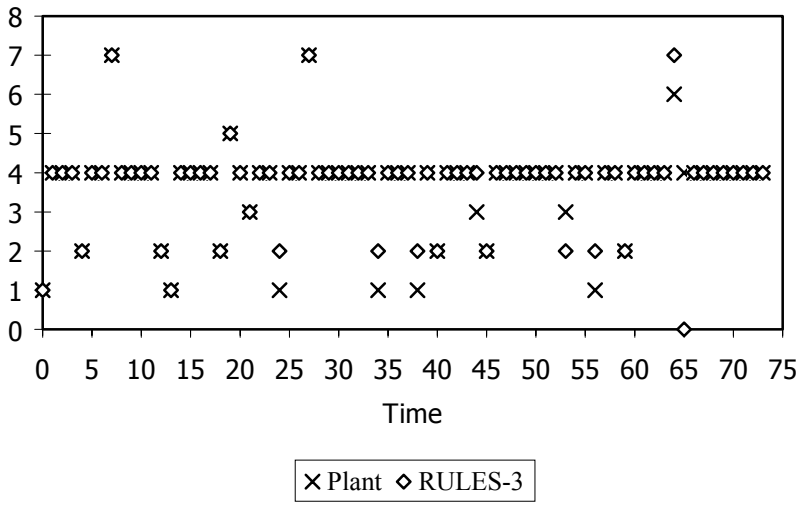

Figure 9. Quantised responses of System 2 and of its model obtained by RULES-3 (number of quantisation levels = 7; number of attributes = 3).

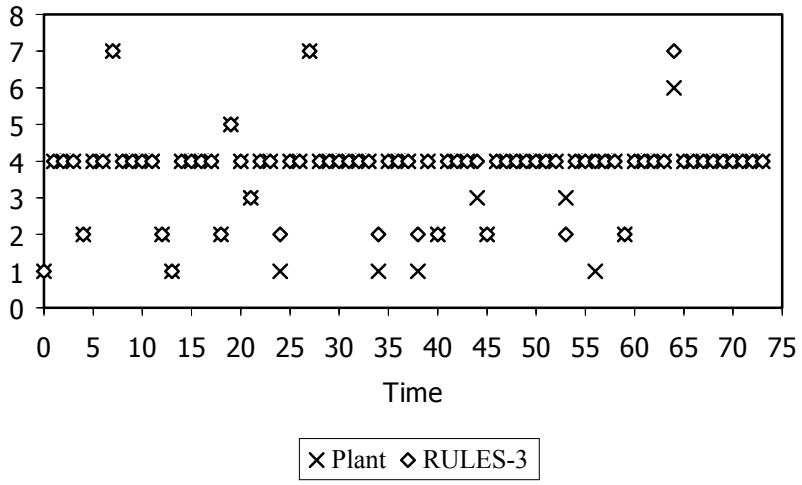

Figure 10. Quantised responses of System 2 and of its model obtained by RULES-3 (number of quantisation levels = 7; number of attributes = 4).

The response of a neural model using an MLP with 2 input units (for y(t) and u(t)), 11 hidden units and 7 output units is charted in Figure 11. Although it employed only relevant attributes, the neural model gave a poorer performance compared to the production rules. The training time for the neural model was 11.4 minutes while the induction time for RULES-3 was 1.92 seconds.

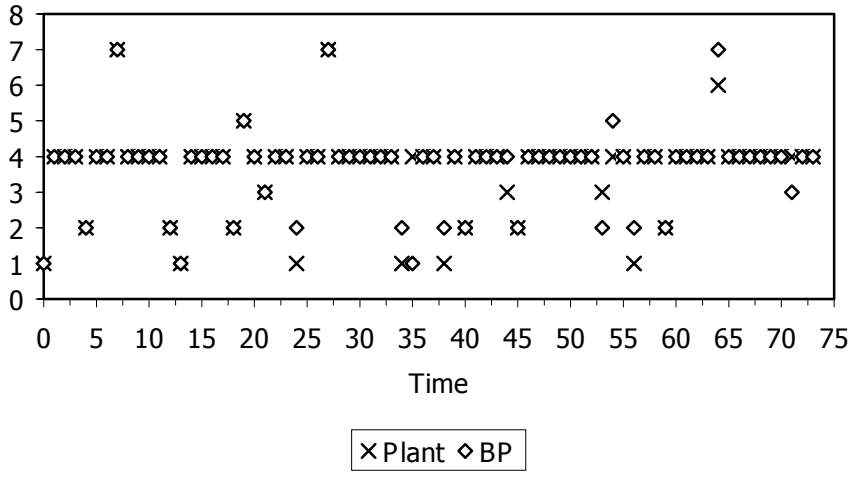

Figure 11. Quantised responses of System 2 and of its model obtained by MLP. (number of quantisation levels = 7; number of attributes = 2).

3.3. System 3

System 3 is a linear third-order plant with the following input-output equation:

$$y(t) = 2.03786\, y(t-1) - 1.366\, y(t-2) + 0.3012\, y(t-3) + 0.003\, u(t-1) + 0.0089\, u(t-2) + 0.000163\, u(t-3)$$

y(t-1), y(t-2), y(t-3), u(t-1), u(t-2) and u(t-3) are the six essential attributes for this problem. 50 quantisation levels were defined. 78 rules were extracted by RULES-3. The quantised responses of the system and the model to a step input are shown in Figure 12. As can be seen, the responses of the model and the system are almost identical.

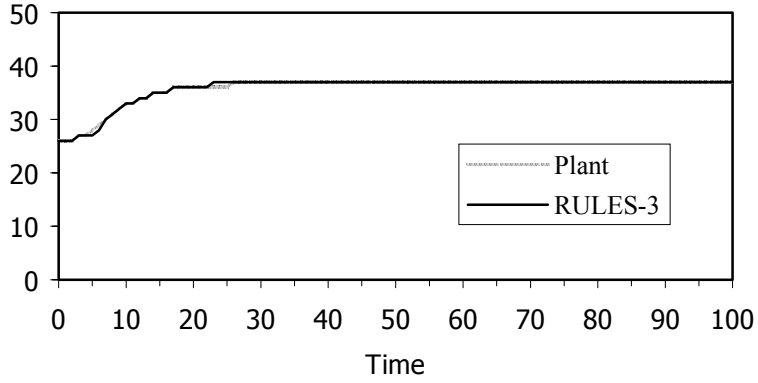

Figure 12. Quantised responses of System 3 and of its model obtained by RULES-3 (number of quantisation levels = 50; number of attributes = 6).

The quantised response of an MLP with 6 input units, 15 hidden units and 13 output units and the response of the plant are plotted in Figure 13. The training time was 1 minute 48 seconds for RULES-3 and 1 hour 49 minutes for the MLP. Note that the number of output units of the MLP was 13 instead of 50 (that is the output

quantisation level was assumed to be 13 rather than 50) to avoid excessive MLP training times.

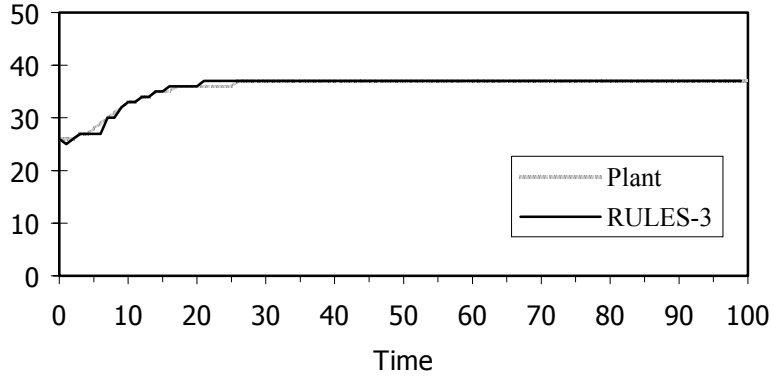

Figure 13. Quantised responses of System 3 and of its model obtained by MLP. (number of quantisation levels = 50; number of attributes = 6).

3.4. System 4

System 4 is a non-linear plant. Its discrete input-output equation is:

$y(t) = 2.03786\ y(t-1) - 0.824\ y(t-2) + 0.130667\ y^3(t-2) - 0.16\ u(t-2)$

y(t-1), y(t-2) and u(t-2) are the three essential attributes for this system. 15 quantisation levels were defined for the system. 25 rules were extracted by RULES-3. The quantised responses of the system and the model to a step input are shown in Figure 14.

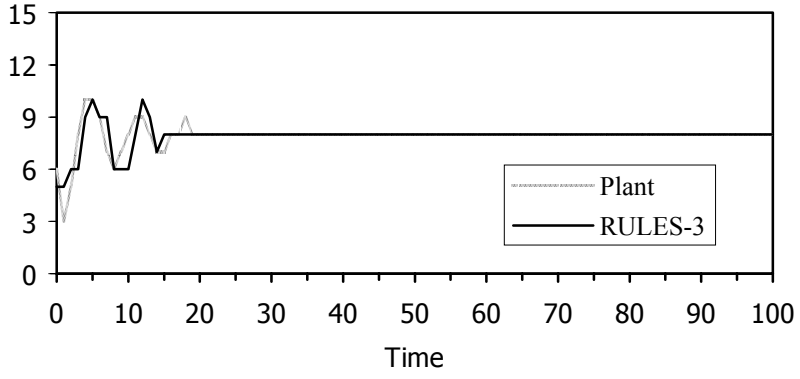

Figure 14. Quantised responses of System 4 and of its model obtained by RULES-3 (number of quantisation levels = 15; number of attributes = 3).

The quantised response of an MLP with 3 input units, 15 hidden units and 7 output units and the response of the plant are shown in Figure 15. The training time was 6.43 seconds for RULES-3 and 29 minutes for the MLP.

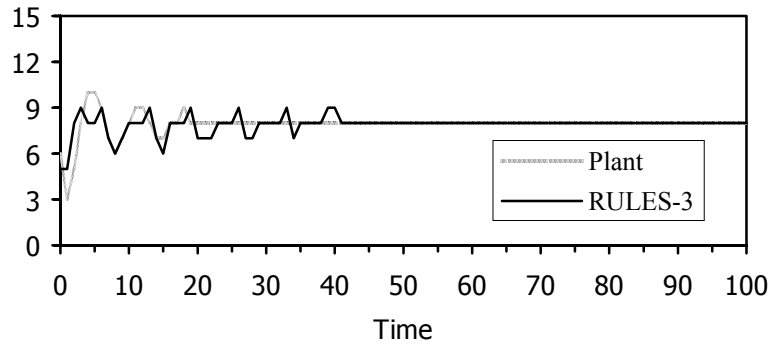

Figure 15. Quantised responses of System 4 and of its model obtained by MLP. (number of quantisation levels = 15; number of attributes = 3).

Table 1 summarises the statistics for the tests on the four systems. Table 2 gives a comparison of the accuracies of models obtained using RULES-3 and MLPs. It can be seen that the former are at least equal to the latter.

Table 1. Summary of statistics for different tests. (QL = number of quantisation levels)

System	No. of attributes	No. of examples	QL	No. of rules	Training time
System 1	4	48	4	16	3.41 s.
System 1	3	29	9	19	2.91 s.
System 1	4	29	9	18	2.52 s.
System 1	5	28	9	16	2.47 s.
System 1	6	27	9	15	2.75 s.
System 2	2	17	7	13	0.99 s.
System 2	3	18	7	12	1.21 s.
System 2	4	19	7	11	1.32 s.
System 3	6	111	50	78	1 min. 48 s.
System 4	3	31	15	25	6.43 s.

Table 2. Comparison of accuracies of RULES-3 and MLP. (QL = number of quantisation levels y_p = output of plant y_m = output of model)

				$\sum \lvert y_p - y_m \rvert$	
	No. of attributes	QL	No. of test examples	RULES-3	MLP
System 1	4	9	100	4	4
System 2	2	7	75	5	5
System 3	6	50	100	6	6
System 4	3	15	100	17	27

4. CONCLUSION

The use of a new induction algorithm to extract models of dynamic systems has been described. Induction-based modelling generally has the advantage of producing explicit models. In addition to this explicitness, the models induced by RULES-3, the algorithm adopted in this work, can have better accuracies than models constructed using neural networks. RULES-3 is easy to apply. It operates efficiently and does not require making assumptions about the structure of the system to be modelled.

5. REFERENCES

1. L. Ljung, *System identification*, Prentice Hall, Englewood Cliffs, NJ, 1987.
2. D.T. Pham and X. Liu, State-space identification of dynamic systems using neural networks, *Eng. Appli. of AI*, V:3, 198-203,1990.
3. D.T. Pham and X. Liu, Dynamic system modelling using partially recurrent neural networks, *Journal of Systems Eng*., 2(2), 90-97,1992.
4. K.S. Narendra and K. Parthasarathy, Identification and control of dynamical systems using neural networks, *IEEE Transactions on Neural Networks*, 1(1), 4-27,1990.
5. S.Z. Qin , H.T. Su, and T.J. McAvoy, Comparison of four neural net learning methods for dynamic system identification, *IEEE Transactions on Neural Network*, 3(1), 122-130,1992.
6. K.J Hunt., D.Sbarbaro, R.Zbikowski and P.J.Gawthrop, Neural networks for control systems- A survey, *Automatica*, 28(6), 1083-1112,1992.
7. S.Chen, S.A. Billings and P.M. Grant, Non-linear system identification using neural networks, *Int. J. Control*, 51(6), 1191-1214,1990.
8. P.M. Mills and A.Y. Zomaya, A neural network approach to on-line identification of non-linear systems, in proc. *IJCNN, Vol. 1*, Singapore, 202-207,1991.
9. N. Bhat and T.J. McAvoy, Use of neural nets for dynamic modeling and control of chemical process systems, *Computers Chem. Eng*., 14(4/5), 573-583,1990.

10. B. Fernandez, A.G. Parlos and W.K. Tsai, Non-linear dynamic system identification using artificial neural networks, *International Joint Conference on Neural Networks, Vol. 2*, June, San Diego, 133-141,1990.
11. A.G. Parlos, A.F. Atiya, K.T. Chong and W.K. Tsai, Non-linear identification of process dynamics using neural networks, *Nuclear Technology*, 97(1), 79-95, 1992.
12. M.S. Aksoy, *New Algorithms for Machine Learning*, Ph.D. thesis, University of Wales, College of Cardiff, U.K, 1994.
13. J.R. Quinlan, C4.*5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, California ISBN 1-55860-238-0, 1993.
14. D.T. Pham and M.S. Aksoy, An algorithm for automatic rule induction, *Artificial Intelligence in Engineering* , No.8, 277-282, 1993.
15. D.T. Pham and M.S. Aksoy, RULES: A simple rule extraction system, *Expert Systems with Applications* , Vol 8, No 1, 59-65, 1995.
16. D.T. Pham D.T. and M.S. Aksoy, A new algorithm for inductive learning, *Journal of Systems Eng*. No. 5, 115-.122., 1995.
17. D.T. Pham and X. Liu, Neural network for discrete dynamic system identification, *Journal of Systems Eng*., 1(1), 51-60, 1991.